Package ‘protGear’

October 17, 2024

Type Package

Title Protein Micro Array Data Management and Interactive Visualization

Version 1.8.0

Description A generic three-step pre-processing package for protein microarray data. This package contains different data pre-processing procedures to allow comparison of their performance.These steps are background correction, the coefficient of variation (CV) based filtering, batch correction and normalization.

License GPL-3

URL https://github.com/Keniajin/protGear

BugReports https://github.com/Keniajin/protGear/issues

Depends R (>= 4.2), dplyr (>= 0.8.0) , limma (>= 3.40.2) ,vsn (>= 3.54.0)

Imports magrittr (>= 1.5) , stats (>= 3.6) , ggplot2 (>= 3.3.0) , tidyr (>= 1.1.3) , data.table (>= 1.14.0), ggpubr (>= 0.4.0), gtools (>= 3.8.2) , tibble (>= 3.1.0) , rmarkdown (>= 2.9) , knitr (>= 1.33), utils (>= 3.6), genefilter (>= 1.74.0), readr (>= 2.0.1) , Biobase (>= 2.52.0), plyr (>= 1.8.6) , Kendall (>= 2.2) , shiny (>= 1.0.0) , purrr (>= 0.3.4), plotly (>= 4.9.0) , MASS (>= 7.3) , htmltools (>= 0.4.0) , flexdashboard (>= 0.5.2) , shinydashboard (>= 0.7.1) , GGally (>= 2.1.2) , pheatmap (>= 1.0.12) , grid(>= 4.1.1), styler (>= 1.6.1) , factoextra (>= 1.0.7) ,FactoMineR (>= 2.4) , rlang (>= 0.4.11), remotes (>= 2.4.0)

Suggests gridExtra (>= 2.3), png (>= 0.1-7) , magick (>= 2.7.3) , ggplotify (>= 0.1.0) , scales (>= 1.1.1) , shinythemes (>= 1.2.0) , shinyjs (>= 2.0.0) , shinyWidgets (>= 0.6.2) , shinycssloaders (>= 1.0.0) , shinyalert (>= 3.0.0) , shinyFiles (>= 0.9.1) , shinyFeedback (>= 0.3.0)

biocViews Microarray, OneChannel, Preprocessing , BiomedicalInformatics , Proteomics , BatchEffect, Normalization , Bayesian, Clustering, Regression,SystemsBiology, ImmunoOncology

Encoding UTF-8

LazyData false

RoxygenNote 7.2.3

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/protGear

git_branch RELEASE_3_19

git_last_commit 512a5f5

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-10-16

Author Kennedy Mwai [cre, aut],
James Mburu [aut],
Jacqueline Waeni [ctb]

Maintainer Kennedy Mwai <keniajin@gmail.com>

Contents

`array_vars` *List the array structure variables*

Description

A generic function returning a list with the data structure.

Usage

```
array_vars(
  channel = "635",
  totsamples,
  FG = "",
  BG = "",
  FBG = "",
  blockspersample,
  chip_path = "data/array_data",
  sampleID_path = "data/array_sampleID/",
  mig_prefix = "_first",
  machine = "",
  date_process = ""
)
```

Arguments

`channel`	A character indicating the channel that the data was scanned at. It is mostly included in the MFI variable names.
`totsamples`	A numeric value indicating teh number of samples on a slide.
`FG`	Optional:A character indicating the name of the foreground variable name. if not specified its created as `paste0("F",channel,".Median")`
`BG`	Optional:A character indicating the name of the background variable name. if not specified its created as `paste0("B",channel,".Median")`
`FBG`	Optional:A character indicating the name of the foreground - background variable name. if not specified its created as `paste0("F",channel,".Median...B",channel)`
`blockspersample`	A numeric value indicating the numer of blocks in a mini-array. The `".gal"` file can help in getting this
`chip_path`	A character indicating the path of the folder location with the array data.
`sampleID_path`	A character indicating the path of the folder location with the sample identifiers matching the array structure.
`mig_prefix`	Optional: A character indicating the identifier of an MIG dilution file

`machine`	Optional:A character indicating the machine used to process the data in the folder
`date_process`	Optional:A character indicating the date when the samples were processed.

Value

a list of parameters required to process the data

genepix_vars

Examples

```
## specify the the parameters to process the data
genepix_vars <- array_vars(
## the channel the data was processed in
  channel = "635",
  ## folder where the array data is stored
  chip_path = "data/array_data",
  ## the number of samples per slide or in as single run
  totsamples = 21,
  ## How many blocks each sample occupies
  blockspersample = 2,
  ## folder where the array data samples id files are stored
  sampleID_path = "data/array_sampleID/",
  ## optional
  mig_prefix = "_first",
  machine = 1,
  date_process = "0520"
)
genepix_vars
```

`best_CV_estimation` *best CV estimation*

Description

A function to select the best CV by combining the replicates in duplicates. The function has been build for up to to 3 replicates so far

Usage

```
best_CV_estimation(dataCV, slide_id, lab_replicates, cv_cut_off)
```

Arguments

`dataCV`	A data frame
`slide_id`	A character string containing the identifier of the data frame variable.
`lab_replicates`	A numeric value indicating the number of lab replicates.
`cv_cut_off`	a numeric value for the CV cut off. Should be between 0-100

Details

Select set of replicates with the best CV

Value

A data frame with the best CV's estimated

Examples

```
dataC <- readr::read_csv(system.file("extdata",
 "dataC.csv", package="protGear"))
## this file has 3 lab replicates and the default names
dataCV <- cv_estimation(dataC  ,lab_replicates=3)
best_CV_estimation(dataCV,slide_id = "iden", lab_replicates = 3,
 cv_cut_off = 20)
```

---

bg_correct *bg_correct*

Description

A generic function to perform background correction.

Usage

```
bg_correct(iden, Data1, genepix_vars, method = "subtract_local")
```

Arguments

`iden`	A character indicating the name of the object to be used under Data1
`Data1`	A data frame with sample identifiers merged with micro array data.
`genepix_vars`	A list of specific definitions of the experiment design. See `array_vars`.
`method`	a description of the background correction to be used. Possible values are `"none"`,`"subtract_local"`,`"subtract_global"`,`"movingmin_bg"`,`"minimum_half"`,`"edwards"` or `"normexp"`. The default is `"subtract_local"`.

Details

Background correction

The function implements background correction methods developed by `backgroundCorrect`. But the `minimum_half` and `movingmin_bg` uses the block of the protein array as the grid. If method="movingmin_bg" the minimum background value within a block is subtracted. If method="minimum_half" then any intensity which is negative after background subtraction is reset to be equal to half the minimum positive value in a block. If method="movingmin_value" then any intensity which is negative after background subtraction is reset to the minimum positive value in a block. For edwards we implement a similar algorithm with `limma::backgroundCorrect(method="edwards")` and for `'normexp'` we use the saddle-point approximation to maximum likelihood, `backgroundCorrect` for more details.

Value

A data frame with background corrected data

`buffer_spots` *Extract buffer spots of data*

Description

A function to extract the buffer spots data. A buffer spot only has the solution for proprietary ingredients for stabilizing protein and minimizing evaporation.

Usage

```
buffer_spots(Data1, buffer_spot = "buffer")
```

Arguments

`Data1`	An object of the class data frame
`buffer_spot`	A character string containing the name of the buffer spots.

Value

A data frame of the buffer control spots

Examples

```
bg_correct_df <- readr::read_csv(system.file("extdata", "Data1_sample.csv",
package="protGear"))
buffer_spots(Data1 = bg_correct_df)
```

`check_sampleID_files` *\\_End_Function_\\ # Check existing sample ID names*

Description

A generic function to check if the file(s) with the MFI values have a corresponding sample ID file. Sample ID file is a file with the identifiers for the samples in array file.

Usage

```
check_sampleID_files(genepix_vars)
```

Arguments

`genepix_vars`	A list of specific definitions of the experiment design. See `array_vars`.

Value

A file with missing corresponding sample ID files

Examples

```
genepix_vars <- array_vars(
channel = "635",
chip_path = system.file("extdata", "array_data/machine1/",
package="protGear"),
totsamples = 21,
blockspersample = 2,
mig_prefix = "_first",
machine = 1,
date_process = "0520"
)
check_sampleID_files(genepix_vars)
```

`create_dir` *Title Create directory function*

Description

creating a directory

Usage

```
create_dir(path)
```

Arguments

`path`	folder location to create a directory

Value

created directory

Examples

```
create_dir("data/sample_folder")
```

`cv_by_sample_estimation` *cv by sample*

Description

A function to give the summary of the CV's by the sampleID

Usage

```
cv_by_sample_estimation(
  dataCV,
  cv_variable,
  lab_replicates,
  sampleID_var = "sampleID"
)
```

Arguments

`dataCV`	A dataframe
`cv_variable`	A character string containing the identifier of the variable with CV values.
`lab_replicates`	A numeric value indicating the number of lab replicates.
`sampleID_var`	A character string containing the name of the sample identifier variable. Default set to 'sampleID'

Details

Summarise CV by samples

Value

A data frame of CV calculated by sample

Examples

```
dataC <- readr::read_csv(system.file("extdata",
"dataC.csv", package="protGear"))
## this file has 3 lab replicates and the default names
dataCV <- cv_estimation(dataC  ,lab_replicates=3)
cv_by_sample_estimation(dataCV, cv_variable = "cvCat_all",
 lab_replicates = 3)
```

cv_estimation *cv_estimation*

Description

A function to calculate the CV for the technical lab replicates. The default values are set as per the object names generated by machine

Usage

```
cv_estimation(
  dataC,
  lab_replicates,
  sampleID_var = "sampleID",
  antigen_var = "antigen",
  replicate_var = "replicate",
  mfi_var = "FMedianBG_correct",
  cv_cut_off = 20
)
```

Arguments

`dataC`	A dataset a data frame with feature variables to be used
`lab_replicates`	A numeric value indicating the number of lab replicates
`sampleID_var`	A character string containing the name of the sample identifier variable. Default set to 'sampleID'
`antigen_var`	A character string containing the name of the features/protein variable. Default to 'antigen'
`replicate_var`	A character string containing the name of the replicate variable. Default to 'replicate'
`mfi_var`	A character string containing the name of the variable with MFI value.Assuming background correction is done already. Default to 'FMedianBG_correct'
`cv_cut_off`	Optional value indicating the cut off of flagging CV's. Default set at 20.

Details

Coefficient of Variation

Value

A data frame where CV's of the replicates have been calculated

Examples

```
dataC <- readr::read_csv(system.file("extdata",
"dataC.csv", package="protGear"))
## this file has 3 lab replicates and the default names
cv_estimation(dataC  ,lab_replicates=3)
```

`error_replicates` \\_*Start_Function_For Error*\\

Description

A generic function to write into the log file with a replicate check error

Usage

```
error_replicates(iden)
```

Arguments

`iden`	An id for the file with replicates error

Value

a log file showing the replicate errors

`extract_bg` *extract bg*

Description

A generic function to extract the background data for micro array data.

Usage

```
extract_bg(iden, data_files, genepix_vars = genepix_vars)
```

Arguments

`iden`	A character indicating the name of the object to be used under data_files.
`data_files`	A list of data objects with names utilised by iden.
`genepix_vars`	A list of specific definitions of the experiment design. See `array_vars`.

Details

Extract the background values

Value

A data frame of background values

Examples

```
## Not run:
genepix_vars <- array_vars(
channel = "635",
chip_path = system.file("extdata", "array_data/machine1/",
package="protGear"),
totsamples = 21,
blockspersample = 2,
mig_prefix = "_first",
machine = 1,
## optional
date_process = "0520"
)
#Define the data path
data_path <- paste0(genepix_vars$chip_path)
# List the file names to use
filenames <- list.files(genepix_vars$chip_path,
                       pattern = '*.txt$|*.gpr$', full.names = FALSE
)
data_files <- purrr::map(
 .x = filenames,
  .f = read_array_files,
  data_path = data_path,
  genepix_vars = genepix_vars
)
data_files <- purrr::set_names(data_files,
purrr::map(filenames, name_of_files))
names(data_files)
extract_bg(iden ="KK2-06" , data_files=data_files,genepix_vars=genepix_vars)
## End(Not run)
```

`launch_protGear_interactive` *launch_protGear_interactive*

Description

This is Function is to launch the shiny application

Usage

```
launch_protGear_interactive()
```

Value

launches the shiny interactive protGear app

Examples

```
app <- system.file("shiny-examples", "protGear_interactive",
"protGear_interactive.Rmd", package = "protGear")
 if (app!=""){
 ## run this
 #launch_protGear_interactive()
 }
```

---

launch_select *launch_select*

Description

This is Function is to launch mutiple shiny applications for protGear

Usage

```
launch_select(theApp)
```

Arguments

theApp	accepts one of the folders containing the shiny appplication

Value

launches the app defined under theApp

Examples

```
validExamples <-
 list.files(system.file("shiny-examples", package = "protGear"))
#launch_select(validExamples[[1]])
```

---

matrix_normalise *Normalize Arrays*

Description

Normalize Arrays

Usage

```
matrix_normalise(
  matrix_antigen,
  method = "log2",
  batch_correct = FALSE,
  batch_var1,
  batch_var2 = day_batches,
  return_plot = FALSE,
  plot_by_antigen = TRUE,
  control_antigens = NULL,
  array_matrix = NULL
)
```

Arguments

`matrix_antigen`	An object of class matrix with features/proteins as columns and samples as the rows
`method`	character string specifying the normalization method. Choices are `"none"`,`"log2"`,`"vsn"`,`"cyclic_loe` `"cyclic_loess_log"` ,`"rlm"`
`batch_correct`	A logical value indicating whether batch correction should be done or not
`batch_var1`	A character or factor vector of size similar to rows of `matrix_antigen` indicating the first batch.
`batch_var2`	A character or factor vector of size similar to rows of `matrix_antigen` indicating the second batch.
`return_plot`	A logical value indicating whether a plot is returned to show the results of normalisation.
`plot_by_antigen`	Logical to indicate whether to plot by antigen or not slide name for the matrix_antigen object.
`control_antigens`	logical vector specifying the subset of spots which are non-differentially-expressed control spots, for use with `method="rlm"`
`array_matrix`	An object of class dataframe or matrix used with `method='rlm'` indicating the sample index and

Value

A data frame of normalised values

Examples

```
matrix_antigen <- readr::read_csv(system.file("extdata",
"matrix_antigen.csv", package="protGear"))
#VSN
normlise_vsn <- matrix_normalise(as.matrix(matrix_antigen),
method = "vsn",
return_plot = TRUE
)
## log
normlise_log <- matrix_normalise(as.matrix(matrix_antigen),
method = "log2",
return_plot = TRUE
)
## cyclic_loess_log
normlise_cylic_log <- matrix_normalise(as.matrix(matrix_antigen),
method = "cyclic_loess_log",
return_plot = TRUE
)
```

merge_sampleID *Merge sample ID with the array data*

Description

A generic function that merges the protein data with the sample identifiers or sample names. If the file does not have sample identifiers the function generates it automatically.

Usage

```
merge_sampleID(iden, data_files, genepix_vars, method)
```

Arguments

iden	A character indicating the name of the object to be used under data_files.
data_files	A list of data objects with names utilised by iden.
genepix_vars	A list of specific definitions of the experiment design. See array_vars.
method	A description of the background correction to be used. See bg_correct.

Value

a data frame merged with corresponding sample ID's. The sample ID are specified in the sample ID files

Examples

```
## Not run:
### Define the genepix_vars
genepix_vars <- array_vars(
  channel = "635",
  chip_path = system.file("extdata", "array_data/machine1/",
   package="protGear"),
  totsamples = 21,
  blockspersample = 2,
  mig_prefix = "_first",
  machine = 1,
  ## optional
  date_process = "0520"
)

## the path where the micro-array data is located
data_path <- paste0(genepix_vars$chip_path)
filenames <- list.files(genepix_vars$chip_path,
                        pattern = "*.txt$|*.gpr$", full.names = FALSE
)
## create a list of all the files
data_files <- purrr::map(
 .x = filenames,
  .f = read_array_files,
  data_path = data_path,
  genepix_vars = genepix_vars
)
data_files <- purrr::set_names(data_files,
purrr::map(filenames, name_of_files))
## merge the lab data with samples and perform bg correction
merge_sampleID(iden = "KK2-06", data_files = data_files,
               genepix_vars =genepix_vars,method = "subtract_global" )
## End(Not run)
```

`minpositive` *Get the minimum positive value*

Description

Get the minimum positive value

Usage

```
minpositive(x)
```

Arguments

`x`	A numeric vector or variable

Value

Returns the minimum positive value in an object

Examples

```
minpositive(c(-1,-2,3,5,6,7,8,9,10))
```

`name_of_files` *Object names of a list*

Description

A generic function returning a vector with the names of files in the same directory. Removes the file extension

Usage

```
name_of_files(i)
```

Arguments

`i`	- a list filenames with .txt or .gpr extension

Value

a list of file names

name

Examples

```
name_of_files("KK2-06.txt")
```

`output_trend_stats` *Trend test using Cox–Stuart (C–S) and Mann–Kendall (M–K) trend tests*

Description

Trend test using Cox–Stuart (C–S) and Mann–Kendall (M–K) trend tests

Usage

```
output_trend_stats(name, p_val, z_val)
```

Arguments

`name`	Name of the test
`p_val`	p value from the test
`z_val`	the Z value of the test

Value

A statistics of mean standard deviation trend

Examples

```
output_trend_stats(name="t.test",p_val=0.001, z_val=5)
```

---

plot_bg *Plot background*

Description

A generic function for plotting of R objects.

Usage

```
plot_bg(df, x_axis = "antigen", bg_MFI = "BG_Median", log_mfi = TRUE)
```

Arguments

`df`	A default dataset to use for plot.
`x_axis`	The variable on the x axis
`bg_MFI`	A numeric `variable` describing which is the background MFI
`log_mfi`	a logical value indicating whether the MFI values should be log transformed or not.

Value

A ggplot of background values

Examples

```
## Not run:
#After extracting the background using \code{\link{extract_bg}}
#we plot the data using
allData_bg <- readr::read_csv(system.file("extdata", "bg_example.csv",
 package="protGear"))
plot_bg(allData_bg,
x_axis = "antigen",
bg_MFI = "BG_Median",  log_mfi = TRUE
)
## End(Not run)
```

plot_buffer *Plot the buffer values*

Description

Plot the buffer values

Usage

```
plot_buffer(
  df = buffers,
  buffer_names = "antigen",
  buffer_mfi = "FMedianBG_correct",
  slide_id = ".id"
)
```

Arguments

df	A data frame to be used to plot
buffer_names	A character string containing the name of the variable with buffer spots. Default set to 'antigen'.
buffer_mfi	A character string containing the name of the variable with MFI value.Assuming background correction is done already. Default to 'FMedianBG_correct'
slide_id	A character string containing the name of the slide/array identifier variable.

Value

plot of buffer spots

Examples

```
buffers <- readr::read_csv(system.file("extdata", "buffers_sample2.csv",
package="protGear"))
plot_buffer(df=buffers,buffer_names = "sampleID")
```

plot_FB *plot_FB*

Description

A generic function for plotting the background and foreground values.

Usage

```
plot_FB(
  df,
  antigen_name = "antigen",
  bg_MFI = "BG_Median",
  FG_MFI = "FBG_Median",
  log_mfi = FALSE
)
```

Arguments

`df`	An object containing the data to which the plot is done.
`antigen_name`	The `variable` describing which features/proteins/ antibodies in the data should be used to plot
`bg_MFI`	A numeric `variable` describing which is the background MFI
`FG_MFI`	A numeric `variable` describing which is the foreground MFI
`log_mfi`	a logical value indicating whether the MFI values should be log transformed or not.

Details

Plot foreground and background values

Value

a ggplot of foreground vs background MFI values

Examples

```
## Not run:
#After extracting the background using \code{\link{extract_bg}}
#we plot the data using
allData_bg <- readr::read_csv(system.file("extdata",
"bg_example.csv", package="protGear"))
plot_FB(allData_bg,
antigen_name = "antigen",
bg_MFI = "BG_Median", FG_MFI = "FBG_Median", log = FALSE
)
## End(Not run)
```

plot_normalised *Comparison of normalised data by sample*

Description

Comparison of normalised data by sample

Usage

```
plot_normalised(exprs_normalised_df, method, batch_correct)
```

Arguments

`exprs_normalised_df`	a normalised data frame
`method`	the method of normalisation used
`batch_correct`	the batch correction

Value

A ggplot of normalised data

Examples

```
matrix_antigen <- readr::read_csv(system.file("extdata",
"matrix_antigen.csv", package="protGear"))
normlise_vsn <- matrix_normalise(as.matrix(matrix_antigen),
method = "vsn",
return_plot = FALSE
)
plot_normalised(normlise_vsn,method="vsn",batch_correct=FALSE)
```

plot_normalised_antigen *Comparison of normalised data by feature*

Description

Comparison of normalised data by feature

Usage

```
plot_normalised_antigen(exprs_normalised_df, method, batch_correct)
```

Arguments

`exprs_normalised_df`	a normalised data frame
`method`	the method of normalisation used
`batch_correct`	the batch correction

Value

A ggplot of various normalisation approaches

Examples

```
matrix_antigen <- readr::read_csv(system.file("extdata",
"matrix_antigen.csv", package="protGear"))
normlise_vsn <- matrix_normalise(as.matrix(matrix_antigen),
method = "vsn",
return_plot = FALSE
)
plot_normalised_antigen(normlise_vsn,method="vsn",batch_correct=FALSE)
```

`read_array_files` *Read array files*

Description

This helps to read the chip file(s).

Usage

```
read_array_files(i, data_path, genepix_vars)
```

Arguments

`i`	The name of the file which the data are to be read from.
`data_path`	The path where the file with the data is located
`genepix_vars`	A list of specific definitions of the experiment design. See `array_vars`.

Details

Read multiple array files

Value

a number of data frames in the global environment

Examples

```
## Not run:
genepix_vars <- array_vars(
channel = "635",
chip_path = system.file("extdata", "array_data/machine1/",
package="protGear"),
totsamples = 21,
blockspersample = 2,
mig_prefix = "_first",
machine = 1,
date_process = "0520"
)
file_read <- "KK2-06.txt"
read_array_files(i=file_read,
data_path=system.file("extdata", "array_data/machine1/",
package="protGear"), genepix_vars=genepix_vars)
## End(Not run)
```

`read_array_visualize` *Read a gpr file to visualize*

Description

Read a gpr file to visualize

Usage

```
read_array_visualize(infile)
```

Arguments

`infile`	a .gpr file to be used to visualize the expression intensities of the slide spots

Value

a data frame to visualize the background or foreground values

Examples

```
## Not run:
read_array_visualize(infile = system.file("extdata",
"/array_data/machine1/KK2-06.txt", package="protGear"))
## End(Not run)
```

`rlm_normalise` *RLM normalisation*

Description

A function for `method='rlm'` from `matrix_normalise`.

Usage

```
rlm_normalise(rlm_normalise_df)
```

Arguments

`rlm_normalise_df`	rlm normalised data frame

Value

an elist of RLM normalisation to be utilised by `rlm_normalise_matrix`

Examples

```
matrix_antigen <- readr::read_csv(system.file("extdata",
"matrix_antigen.csv", package="protGear"))
#rlm_normalise_df <- rlm_normalise_matrix(matrix_antigen=matrix_antigen,
#array_matrix=array_matrix,
# control_antigens=control_antigens)
# rlm_normalise(rlm_normalise_df)
```

`rlm_normalise_matrix` *Nomrmalise using RLM*

Description

A function for `method='rlm'` from `matrix_normalise`.

Usage

```
rlm_normalise_matrix(matrix_antigen, array_matrix, control_antigens)
```

Arguments

`matrix_antigen`	A matrix with antigen data
`array_matrix`	A matrix with control antigen data
`control_antigens`	the control antigens for RLM normalisation

Value

A RLM normalised data frame

Examples

```
matrix_antigen <- readr::read_csv(system.file("extdata",
 "matrix_antigen.csv", package="protGear"))
# rlm_normalise_matrix(matrix_antigen=matrix_antigen,
 #array_matrix=array_matrix,
# control_antigens=control_antigens)
```

`tag_subtract` *tag_subtract*

Description

\\_End_Function_\\ #

Usage

```
tag_subtract(
  dataC_mfi,
  tag_antigens,
  mean_best_CV_var,
  tag_file,
  batch_vars,
  sampleID_var = "sampleID",
  antigen_var = "antigen"
)
```

Arguments

`dataC_mfi`	A dataframe
`tag_antigens`	A character vector with the names of proteins or antigens used as TAG.
`mean_best_CV_var`	A character string containing the identifier of the variable with the MFI values.
`tag_file`	A data frame with variables `antigen`, `TAG`, `TAG_name` to show the TAG for the different antigens or proteins in dataC_mfi
`batch_vars`	A list of characters identifying variables in dataC_mfi for indicating batch.
`sampleID_var`	A character string containing the name of the sample identifier variable. Default set to 'sampleID'
`antigen_var`	A character string containing the name of the features/protein variable. Default to 'antigen'

Details

Subtract the purification TAG data

Value

A data frame of TAG values subtracted

Examples

```
tag_file <- readr::read_csv(system.file("extdata", "TAG_antigens.csv",
package="protGear"))
tag_antigens <- c("CD4TAG", "GST", "MBP")
batch_vars <- list(machine = "m1", day = "0520")
dataC <- readr::read_csv(system.file("extdata", "dataC.csv",
 package="protGear"))
## this file has 3 lab replicates and the default names
dataCV <- cv_estimation(dataC  ,lab_replicates=3)
dataCV_best2 <- best_CV_estimation(dataCV,slide_id = "iden",
lab_replicates = 3, cv_cut_off = 20)
tag_subtract(dataCV_best2,tag_antigens=tag_antigens,
mean_best_CV_var="mean_best_CV",
 tag_file = tag_file,antigen_var = "antigen", batch_vars = batch_vars)
```

`visualize_slide` *Visualize the slide mimicking the original scan image.*

Description

Visualize the slide mimicking the original scan image.

Usage

```
visualize_slide(infile, MFI_var, interactive = FALSE, d_f = NA)
```

Arguments

`infile`	a .gpr file to be used to visualize the expression intensities of the slide spots
`MFI_var`	the MFI variable to plot, can be either the background or foreground value
`interactive`	a logical to specify whether an interactive graph is returned or not
`d_f`	a data frame with array data

Value

A ggplot of slide foreground values

Examples

```
## Not run:
visualize_slide(
infile = system.file("extdata", "/array_data/machine1/KK2-06.txt",
 package="protGear"),
MFI_var = "B635 Median"
)
## End(Not run)
```

---

visualize_slide_2d *Visualize the slide mimicking the original scan image using a 2d plot.*

Description

Visualize the slide mimicking the original scan image using a 2d plot.

Usage

```
visualize_slide_2d(infile, MFI_var, d_f = NA)
```

Arguments

`infile`	- a .gpr file to be used to visualize the expression intensities of the slide spots
`MFI_var`	the MFI variable to plot, can be either the background or foreground value
`d_f`	a data frame with array data

Value

A 2d plot of either the background or foreground values

Examples

```
## Not run:
visualize_slide_2d(
infile = system.file("extdata", "/array_data/machine1/KK2-06.txt",
package="protGear"),
MFI_var = "B635 Median"
)
## End(Not run)
```

Index